

# Feature Selection Techniques in Machine Learning: From Intuition to Practical Methods

Ahmed BADI  
[ahmedbadi905@gmail.com](mailto:ahmedbadi905@gmail.com)  
[linkedin.com/in/badi-ahmed](https://linkedin.com/in/badi-ahmed)

20 December 2025

## Abstract

Feature selection is a fundamental step in machine learning that focuses on choosing the most informative input variables for a model. Instead of feeding all available features to an algorithm, we select a smaller and more relevant subset. This can improve accuracy, reduce overfitting, speed up training, and make models easier to understand. In this article, we introduce the core ideas behind feature selection, explain why it matters, and present three main families of methods: filter, wrapper, and embedded techniques. For each family, we give intuitive explanations, mathematical formulations, and practical examples, including correlation-based filters, information-theoretic measures, stepwise selection, and regularization-based methods such as LASSO. We also discuss how to evaluate feature subsets and highlight common pitfalls and best practices in real projects.

**Keywords:** Feature Selection, Dimensionality Reduction, Filter Methods, Wrapper Methods, Embedded Methods, Mutual Information, Information Gain, LASSO, Model Interpretability.

## 1 Introduction

Modern datasets often contain tens, hundreds, or even thousands of features. On paper, more features should give a model more information. In practice, extra features can introduce noise, redundancy, and spurious correlations. This can hurt model performance instead of helping it. [1] Feature selection is the process of choosing a subset of relevant features that are most useful for predicting the target variable.

Feature selection is different from feature extraction. In feature extraction, we create new transformed features (for example using PCA). In feature selection, we keep a subset of the *original* features and drop the others. This is attractive because selected features remain directly interpretable in the original domain. [2]

Feature selection helps in several ways:

- **Better generalization:** Removing irrelevant features can reduce overfitting.
- **Faster training and prediction:** Fewer features often mean less computation.
- **Improved interpretability:** A smaller set of meaningful features is easier to explain to domain experts.
- **Noise reduction:** Irrelevant or redundant signals are filtered out before modeling.

In this article, we present a structured overview of feature selection techniques, focusing on three main categories that are widely used in practice: filter methods, wrapper methods, and embedded methods. [3], [4], [5]

## 2 Problem Setup and Notation

We consider a supervised learning setting with a dataset

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n,$$

where each input vector  $\mathbf{x}_i \in \mathbb{R}^p$  has  $p$  features, and  $y_i$  is the target (continuous for regression or discrete for classification).

We denote the full feature set by

$$\mathcal{F} = \{1, 2, \dots, p\},$$

and a candidate subset of features by  $S \subseteq \mathcal{F}$ . Feature selection aims to find a “good” subset  $S^*$  according to some objective:

$$S^* = \arg \max_{S \subseteq \mathcal{F}} J(S),$$

where  $J(S)$  measures the quality of subset  $S$ . Examples include:

- Predictive performance (accuracy, F1, RMSE) of a model trained on features  $S$ .
- Statistical relevance measures such as correlation or mutual information between  $X_S$  and  $y$ .
- A trade-off between performance and subset size, for example via regularization.

Because the number of possible subsets grows as  $2^p$ , exhaustive search is usually impossible for realistic  $p$ . Feature selection methods therefore approximate  $S^*$  using heuristics and different search strategies. [1]

## 3 Types of Feature Selection Methods

Feature selection methods are commonly grouped into three main families: **filter**, **wrapper**, and **embedded** methods. [3], [4], [5]

- **Filter methods** evaluate features using simple statistics, independently of any particular model.
- **Wrapper methods** use a predictive model to score subsets of features based on their performance.
- **Embedded methods** perform feature selection inside the model training process itself.

These categories are summarized in Table 1.

Method Type	Main Idea	Examples	Cost
Filter	Rank or select features using statistical scores, independent of the model	Correlation, IG, chi-square, mutual information	Low
Wrapper	Search over feature subsets using model performance as criterion	Forward selection, backward elimination, RFE	High
Embedded	Selection happens during model training via penalties or built-in importance	LASSO, Elastic Net, tree-based importance	Medium

Table 1: Types of feature selection methods.

In the following sections, we examine each family in more detail.

## 4 Filter Methods

Filter methods use generic statistical measures to evaluate how strongly each feature is related to the target. They do not depend on a specific machine learning algorithm. [4], [5]

### 4.1 Information-based Metrics: IG and MI

Information-based metrics quantify how much knowing a feature reduces uncertainty about the target. Two popular measures are **Information Gain** and **Mutual Information**. [6]

**Entropy and Information Gain.** Entropy of a target variable  $Y$  with possible values  $\{y_1, \dots, y_K\}$  is

$$H(Y) = - \sum_{k=1}^K p(y_k) \log_2 p(y_k).$$

If we split the dataset on a feature  $A$ , we obtain subsets  $S_v$  for each value  $v$  of  $A$ . The Information Gain (IG) of splitting  $S$  on attribute  $A$  is

$$IG(S, A) = H(S) - \sum_v \frac{|S_v|}{|S|} H(S_v).$$

Higher IG means the feature reduces more uncertainty in the target. Decision tree algorithms such as ID3 and C4.5 use IG (or its variants) to choose split attributes. [6]

**Mutual Information.** Mutual Information (MI) between a feature  $X$  and target  $Y$  is

$$I(X; Y) = \sum_{x,y} p(x, y) \log \frac{p(x, y)}{p(x) p(y)}.$$

It measures how much knowing  $X$  reduces uncertainty about  $Y$ , and can capture non-linear dependencies. [7], [8]

### 4.2 Information Gain Ratio

Information Gain can be biased toward attributes with many distinct values. Information Gain Ratio (IGR) normalizes IG by the entropy of the attribute  $A$ : [6]

$$IGR(S, A) = \frac{IG(S, A)}{H(A)},$$

where  $H(A)$  measures how widely the attribute splits the data. Higher IGR indicates a feature that both reduces uncertainty and does not simply over-partition the data.

### 4.3 Common Filter Techniques

Below we summarize common filter techniques you listed, with short explanations.

**Chi-square Test.** The chi-square ( $\chi^2$ ) test checks whether there is a significant association between a categorical feature and a categorical target. For a contingency table with observed counts  $O_{ij}$  and expected counts  $E_{ij}$ ,

$$\chi^2 = \sum_i \sum_j \frac{(O_{ij} - E_{ij})^2}{E_{ij}}.$$

Features with high  $\chi^2$  values (and low p-values) are considered more dependent on the target. [9]

**Fisher's Score.** Fisher's score evaluates how well a feature separates classes. For a binary problem with class means  $\mu_1, \mu_2$  and variances  $\sigma_1^2, \sigma_2^2$ ,

$$F = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}.$$

Higher values mean better class separation. This idea generalizes to multi-class cases. [10]

**Pearson's Correlation Coefficient.** For continuous feature  $X$  and continuous target  $Y$ ,

$$\rho(X, Y) = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}.$$

$|\rho|$  close to 1 indicates a strong linear relationship; features with very low  $|\rho|$  may be less useful in linear models. [11]

**Variance Threshold.** Very low-variance features carry little information because they are almost constant. A simple filter removes features with variance below a threshold:

$$\text{Var}(X_j) = \frac{1}{n} \sum_{i=1}^n (x_{ij} - \bar{x}_j)^2.$$

If  $\text{Var}(X_j) < \tau$ , feature  $j$  is dropped. [12]

**Mean Absolute Difference (Mean Absolute Deviation).** Instead of squared deviations, mean absolute deviation uses absolute differences:

$$\text{MAD}(X_j) = \frac{1}{n} \sum_{i=1}^n |x_{ij} - \bar{x}_j|.$$

Low MAD indicates a feature that does not vary much; such features can also be removed. [13]

**Dispersion Ratio.** The dispersion ratio is sometimes defined as the ratio between arithmetic mean and geometric mean:

$$DR(X_j) = \frac{\bar{x}_j}{(\prod_{i=1}^n x_{ij})^{1/n}},$$

for positive features. A higher ratio can indicate greater dispersion and potential discriminative power. [14]

#### 4.4 Summary of Filter Techniques

Table 2 summarizes the main filter methods discussed.

Technique	What it measures	Typical use
Information Gain / IGR	Reduction in entropy after splitting on a feature	Decision trees, discrete features
Mutual Information	General dependency (linear and non-linear)	Feature relevance in classification/regression
Chi-square test	Association between categorical feature and target	Categorical data, text features
Fisher's Score	Class separability for a feature	Binary/multi-class problems
Pearson Correlation	Linear relationship between feature and target	Regression, linear models
Variance / MAD	Amount of variation of a feature	Remove almost-constant features
Dispersion ratio	Relative dispersion via mean vs geometric mean	Positive-valued features (e.g. intensities)

Table 2: Common filter-based feature selection techniques.

## 4.5 Advantages and Limitations of Filter Methods

### Advantages:

- Very fast and scalable to high dimensional data.
- Independent of any specific model, which makes them reusable.
- Less risk of overfitting than wrapper methods because they use simple statistics.

### Limitations:

- Evaluate each feature mostly in isolation, ignoring interactions between features.
- A feature that is weak alone but strong in combination with others may be discarded.

## 5 Wrapper Methods

Wrapper methods treat feature selection as a search problem: they repeatedly train and evaluate a model on different subsets of features, using predictive performance as the objective. [1], [5]

### 5.1 Forward Selection

**Forward Selection** starts from an empty feature set and adds features one by one. [15]

1. Initialize  $S_0 = \emptyset$ .
2. At step  $t$ , consider adding each remaining feature  $j \notin S_{t-1}$ . For each candidate subset  $S_{t-1} \cup \{j\}$ , train a model and compute performance  $J(S)$  (e.g. cross-validated accuracy).
3. Choose

$$j^* = \arg \max_{j \notin S_{t-1}} J(S_{t-1} \cup \{j\}),$$

and set  $S_t = S_{t-1} \cup \{j^*\}$ .

4. Stop when adding features does not improve  $J(S)$  or when you reach a target number of features.

## 5.2 Backward Elimination

**Backward Elimination** starts from all features and removes the least useful ones. [16]

1. Initialize  $S_0 = \mathcal{F}$ .
2. At step  $t$ , for each feature  $j \in S_{t-1}$ , evaluate performance  $J(S_{t-1} \setminus \{j\})$ .
3. Remove the feature whose removal hurts performance the least:

$$j^* = \arg \max_{j \in S_{t-1}} J(S_{t-1} \setminus \{j\}),$$

and set  $S_t = S_{t-1} \setminus \{j^*\}$ .

4. Stop when further removals reduce performance too much or when a minimal subset size is reached.

## 5.3 Recursive Feature Elimination (RFE)

**Recursive Feature Elimination (RFE)** is another popular wrapper method. Instead of adding features from scratch, RFE uses a model that provides feature importance scores (for example, linear models or tree-based models). [17]

1. Train a model on the full feature set.
2. Compute feature importance (for example, absolute coefficient values or impurity-based importance).
3. Remove one or a few least important features.
4. Retrain the model on the reduced feature set and repeat until the desired number of features remains.

## 5.4 Objective Function

Wrapper methods define a quality function  $J(S)$  based on model performance, for example:

$$J(S) = \text{Accuracy}_{\text{CV}}(S),$$

where  $\text{Accuracy}_{\text{CV}}(S)$  is the cross-validated accuracy of a classifier trained using features in  $S$ .

In regression, one might use negative mean squared error:

$$J(S) = -\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2.$$

## 5.5 Advantages and Limitations of Wrapper Methods

### Advantages:

- Directly optimize what we care about: model performance.
- Can capture interactions between features, not just individual relevance.

### Limitations:

- Computationally expensive, especially when  $p$  is large.
- Higher risk of overfitting, since the same data is used repeatedly for selection and evaluation if cross-validation is not handled carefully.

## 6 Embedded Methods

Embedded methods integrate feature selection directly into the model training process. The model itself decides which features are important while fitting the data. [4], [18], [19]

### 6.1 LASSO: L1-regularized Regression

Consider linear regression with parameters  $\beta \in \mathbb{R}^p$ . The standard least squares objective is

$$\min_{\beta} \frac{1}{2n} \sum_{i=1}^n (y_i - \beta^\top \mathbf{x}_i)^2.$$

The LASSO (Least Absolute Shrinkage and Selection Operator) adds an  $L_1$  penalty to encourage sparsity:

$$\min_{\beta} \frac{1}{2n} \sum_{i=1}^n (y_i - \beta^\top \mathbf{x}_i)^2 + \lambda \|\beta\|_1,$$

where

$$\|\beta\|_1 = \sum_{j=1}^p |\beta_j|$$

and  $\lambda \geq 0$  controls the strength of regularization. [20]

For large enough  $\lambda$ , some coefficients  $\beta_j$  become exactly zero. Features with  $\beta_j = 0$  are effectively removed from the model, so LASSO naturally performs feature selection.

### 6.2 Logistic Regression with L1 Penalty

For binary classification ( $y_i \in \{0, 1\}$ ), logistic regression models

$$P(Y = 1 | \mathbf{x}) = \sigma(\beta^\top \mathbf{x}) = \frac{1}{1 + e^{-\beta^\top \mathbf{x}}}.$$

The regularized loss with  $L_1$  penalty becomes

$$\min_{\beta} -\frac{1}{n} \sum_{i=1}^n \left[ y_i \log \sigma(\beta^\top \mathbf{x}_i) + (1 - y_i) \log(1 - \sigma(\beta^\top \mathbf{x}_i)) \right] + \lambda \|\beta\|_1.$$

Again, the  $L_1$  penalty can set some coefficients to zero, selecting a subset of features directly as part of model fitting. [19]

### 6.3 Tree-based Methods and Feature Importance

Tree-based models such as decision trees, random forests, and gradient boosting machines provide a different type of embedded feature selection. [18], [21]

Each split in a tree is based on a feature and a threshold. The quality of a split is measured by impurity decrease. The overall importance of feature  $j$  can be computed as

$$\text{Importance}(j) = \sum_{t \in \mathcal{T}_j} \Delta I_t,$$

where  $\mathcal{T}_j$  is the set of all nodes where feature  $j$  is used for splitting, and  $\Delta I_t$  is the impurity reduction at node  $t$ . Features with low importance scores can be removed after training, or only the top  $k$  features can be kept.

## 6.4 Advantages and Limitations of Embedded Methods

### Advantages:

- More efficient than wrapper methods, since selection and training happen together.
- Often give better predictive performance than pure filters, because they consider the learning algorithm.
- Natural in regularized linear models and tree-based ensembles widely used in practice. [18]

### Limitations:

- Model-dependent: the selected features may not be optimal for a different type of model.
- Some methods (for example LASSO) require careful tuning of the regularization parameter  $\lambda$ .

## 7 Practical Workflow and Tips

In practice, feature selection is part of a larger machine learning pipeline. A typical workflow is:

1. **Data cleaning and preprocessing:** Handle missing values, encode categorical variables, and scale numerical features when necessary.
2. **Initial filter step:** Remove obviously irrelevant features using simple filters (low variance, low correlation, chi-square or IG).
3. **Model-based selection:** Use embedded methods such as LASSO or tree-based models to further refine the feature set.
4. **Optional wrapper refinement:** Apply a wrapper method (for example, RFE) with cross-validation to squeeze out extra performance.
5. **Evaluation:** Compare models with and without feature selection using cross-validation and holdout sets to ensure that selection truly helps generalization.

Some practical recommendations:

- Always perform feature selection inside cross-validation loops to avoid optimistic bias.
- Combine domain knowledge with automatic methods: experts often know which features are meaningful.
- Be careful with highly correlated features: LASSO may arbitrarily select among them; tree-based models may split on one but ignore another.

## 8 Conclusion

Feature selection is a powerful tool for simplifying models, improving generalization, and making machine learning solutions more interpretable. In this article, we have presented:

- The basic problem setup and goals of feature selection.
- Filter methods based on statistics such as Information Gain, mutual information, chi-square, correlation, and simple dispersion measures.

- Wrapper methods that search over feature subsets using model performance, including forward selection, backward elimination, and RFE.
- Embedded methods that integrate selection into model training, including LASSO and tree-based feature importance.

There is no single “best” technique that works for all problems. The right approach depends on the size of the dataset, the number of features, the type of model, and the available computational budget. In real projects, it is common to combine several techniques and to validate choices carefully using cross-validation and domain knowledge. [1], [3]

## References

- [1] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.
- [2] E. Alpaydin, *Introduction to Machine Learning*. MIT Press, 2014.
- [3] S. Raschka, *What is the difference between filter, wrapper, and embedded feature selection?* [https://sebastianraschka.com/faq/docs/feature\\_sele\\_categories.html](https://sebastianraschka.com/faq/docs/feature_sele_categories.html), 2025.
- [4] A. Izen, *Feature selection: Filter method, wrapper method and embedded method*, <https://izen.ai/2019/09/04/feature-selection-filter-method-wrapper-method-and-embedded-method/>, 2019.
- [5] GeeksforGeeks, *Feature selection techniques in machine learning*, <https://www.geeksforgeeks.org/machine-learning/feature-selection-techniques-in-machine-learning/>, 2021.
- [6] GeeksforGeeks, *Information gain and mutual information for machine learning*, <https://www.geeksforgeeks.org/machine-learning/information-gain-and-mutual-information-for-machine-learning/>, 2025.
- [7] T. Huijskens, *Mutual information-based feature selection*, <https://thuijskens.github.io/2017/10/07/feature-selection/>, 2017.
- [8] scikit-learn developers, *Mutual\_info\_classif*, [https://scikit-learn.org/stable/modules/generated/sklearn.feature\\_selection.mutual\\_info\\_classif.html](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.mutual_info_classif.html), 2013.
- [9] GeeksforGeeks, *Chi-square test*, <https://www.geeksforgeeks.org/mathematics/chi-square-test/>, 2024.
- [10] GeeksforGeeks, *Fisher’s f-test in R programming*, <https://www.geeksforgeeks.org/r-language/fishers-f-test-in-r-programming/>, 2023.
- [11] GeeksforGeeks, *Pearson correlation coefficient*, <https://www.geeksforgeeks.org/mathematics/pearson-correlation-coefficient/>, 2022.
- [12] GeeksforGeeks, *Variance threshold in machine learning*, <https://www.geeksforgeeks.org/machine-learning/variance-threshold/>, 2020.
- [13] GeeksforGeeks, *Mean absolute deviation*, <https://www.geeksforgeeks.org/mathematics/mean-absolute-deviation/>, 2021.
- [14] GeeksforGeeks, *Measures of dispersion*, <https://www.geeksforgeeks.org/mathematics/measures-of-dispersion/>, 2019.
- [15] GeeksforGeeks, *Forward feature selection in machine learning*, <https://www.geeksforgeeks.org/machine-learning/forward-feature-selection-in-machine-learning/>, 2023.
- [16] GeeksforGeeks, *Backward elimination technique in multiple linear regression*, <https://www.geeksforgeeks.org/machine-learning/ml-multiple-linear-regression-backward-elimination-technique/>, 2020.

- [17] GeeksforGeeks, *Recursive feature elimination*, <https://www.geeksforgeeks.org/machine-learning/recursive-feature-elimination/>, 2022.
- [18] GeeksforGeeks, *Feature selection / embedded methods*, <https://www.geeksforgeeks.org/machine-learning/feature-selection-embedded-methods/>, 2025.
- [19] S. Raschka, *L1-regularized logistic regression as embedded feature selection*, [https://www.youtube.com/watch?v=\\_aGWjt7GKBE](https://www.youtube.com/watch?v=_aGWjt7GKBE), 2021.
- [20] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society: Series B*, vol. 58, pp. 267–288, 1996.
- [21] P. Geurts, D. Ernst, and L. Wehenkel, “Extremely randomized trees,” *Machine Learning*, vol. 63, no. 1, pp. 3–42, 2006.